Multilevel Approximate Robust Principal Component Analysis

Vahan Hovhannisyan Yannis Panagakis Stefanos Zafeiriou
Panos Parpas
Imperial College London, UK
{v.hovhannisyan13, i.panagakis, s.zafeiriou, p.parpas}@imperial.ac.uk

Abstract

Robust principal component analysis (RPCA) is currently the method of choice for recovering a low-rank matrix from sparse corruptions that are of unknown value and support by decomposing the observation matrix into low-rank and sparse matrices. RPCA has many applications including background subtraction, learning of robust subspaces from visual data, etc. Nevertheless, the application of SVD in each iteration of optimisation methods renders the application of RPCA challenging in cases when data is large. In this paper, we propose the first, to the best of our knowledge, multilevel approach for solving convex and non-convex RPCA models. The basic idea is to construct lower dimensional models and perform SVD on them instead of the original high dimensional problem. We show that the proposed approach gives a good approximate solution to the original problem for both convex and non-convex formulations, while being many times faster than original RPCA methods in several real world datasets.

1. Introduction

Low-rank matrix recovery is a cornerstone in data analysis and dimensionality reduction, with the principal component analysis (PCA) [12] being the most widely employed method for this task. Even though the PCA is easy to do by means of eigen-decomposition, it is fragile to the presence of gross, non-Gaussian noise and outliers and the estimated low-rank subspace may be arbitrarily away from the true one; even when a small fraction of the data is corrupted [14]. To alleviate this, robust PCA (RPCA) models have been proposed [6]. The RPCA aims to recover a low-rank matrix from sparse corruptions that are of unknown value and support by decomposing the observation matrix ($\mathbf{D}$) into two terms: a low-rank matrix ($\mathbf{L}$) and a sparse one ($\mathbf{S}$) , accounting for sparse noise and outliers, namely $\mathbf{D} = \mathbf{L} + \mathbf{S}$. This model has profound impact in visual data analysis and computer vision applications such as image denoising [6], background substraction, image alignment [25], texture recovery [30], deformable models [26], face frontalization [27], structure from motion [2], to mention but a few examples.

A natural approach to estimate the low-rank plus sparse decomposition of the RPCA is to minimize the rank of $\mathbf{L}$ and the number on non-zero entries of $\mathbf{S}$, measured by ℓ_0 quasi norm [6]. Unfortunately, both rank and ℓ_0-norm minimization is NP-hard [29, 21]. The nuclear- and the ℓ_1-norms are typically adopted as convex surrogates to rank and ℓ_0- norm, respectively yielding a convex relaxation, which can be provably solved under some natural conditions on the low rank and sparse components. Common solvers for the convex RPCA include: Iterative Thresholding (IT) [8], Accelerated Proximal Gradient (APG) [24], Augmented Lagrange Multipliers (ALM) and Augmented Lagrangian Alternating Direction method [16]. However, the above mentioned solvers exhibit significant computational drawbacks. In particular, the convex RPCA model can be solved with at most $\mathcal{O}(1/\epsilon)$ iterations, where ϵ is the solution accuracy, each iteration requires a singular value decomposition (SVD), which can be computationally expensive even when only a few singular values are calculated. Moreover, the $\mathcal{O}(1/\epsilon)$ convergence rate is much slower that the $\mathcal{O}(log(1/\epsilon))$ rate of the classical PCA.

Recent advances in non-convex optimization enable the development of algorithms that partially alleviate the computational burden of convex RPCA. Concretely, Netrapalli et al [22] proposed to solve a non-convex problem of finding a low rank plus sparse matrix decomposition, by means of alternating projections onto non-convex sets. Surprisingly, the method provably converges to a unique minimiser with linear rate $\mathcal{O}(\log(1/\epsilon))$, which is much faster that the sublinear rate of convex methods. However, for large problems (even partial SVDs) can require unacceptably long time to solve the problem. There have been several attempts to reduce the computational time of large nuclear norm regularized optimization problems. Namely, [17] proposed to reduce the problem dimension by writing the large solution matrix as a product of a small orthogonal and another matrix. They solved the resulting non-convex problem via an

augmented Lagrangian alternating direction method. Another very popular approach for reducing large problem dimensions is to create smaller sub-problems using various randomized techniques [18, 23, 9, 19, 1].

In this paper, motivated by the recent advances in multilevel optimization algorithms [20, 13] we propose a simple, yet quite generic and very effective approach for significantly reducing computational costs for many problems requiring low rank matrix approximations, including convex and non-convex robust PCA models. We exploit the fact that many problems arising from computer vision and machine learning applications can be modelled using various degrees of fidelity. For instance, video frames from a fixed camera or facial images taken with varying illuminations are highly correlated and therefore their linear combinations maintain the model's underlying information. The basic idea is to construct and solve lower dimensional (coarse) models for each subproblem and then prolong its solution to the original problem dimension. We show that using special restriction and prolongation operators ensure good low rank approximations. Then we demonstrate our proposed multilevel low rank approximation technique within both convex and non-convex models. As several video background extraction and facial shadow removal experiments show, our multilevel approach can speed up its original method by several times.

Notations. Throughout the paper, scalars are denoted by lower-case letters, vectors (matrices) are denoted by lower-case (upper-case) boldface letters, e.g. $\mathbf{x}$ ($\mathbf{X}$). $\mathbf{I}$ denotes the identity matrix with appropriate dimension. The ℓ_1 and ℓ_2 norms of a vector $\mathbf{x}$ are defined as $\|\mathbf{x}\|_1 = \sum_i |x_i|$ and $\|\mathbf{x}\|_2 = \sqrt{\sum_i x_i^2}$, respectively. The matrix ℓ_1 norm is defined as $\|\mathbf{X}\|_1 = \sum_i \sum_j |x_{ij}|$, where $|\cdot|$ denotes the absolute value operator. The Frobenius norm is defined as $\|\mathbf{X}\|_F = \sqrt{\sum_i \sum_j x_{ij}^2}$, and the nuclear norm of $\mathbf{X}$ (i.e., the sum of singular values of a matrix) is denoted by $\|\mathbf{X}\|_*$. The l-th largest (in absolute value) singular value of matrix $\mathbf{X}$ is denoted as $\sigma_l(\mathbf{X})$. In algorithm pseudocodes we use $\mathbf{X}^{(k)}$ (u_k) to denote the value of matrix $\mathbf{X}$ (scalar u) at iteration k.

2. RPCA Methods

In this section, we present the state of the art methods for solving the convex and non-convex robust PCA problems. Note that for both methods, the computational bottleneck are the SVDs requiring $\mathcal{O}(rmn)$ operations each, where r is the number of required singular values.

2.1. Inexact ALM for RPCA

The problem of representing an input data matrix $\mathbf{D} \in \mathbb{R}^{m\times n}$ as a sum of a low rank matrix $\mathbf{L}^\star$ and a sparse matrix $\mathbf{S}^\star$ can be exactly solved via the following convex optimization problem:

$$\min_{\mathbf{L},\mathbf{S}} \|\mathbf{L}\|_* + \lambda\|\mathbf{S}\|_1, \quad \textit{subject to} \quad \mathbf{D} = \mathbf{L} + \mathbf{S}, \tag{1}$$

where $\lambda > 0$ is a weighting parameter. A classical approach for solving (1) is by minimizing its augmented Lagrangian defined as

$$\begin{aligned}\mathcal{L}(\mathbf{L},\mathbf{S},\mathbf{Y},\mu) = \|\mathbf{L}\|_* + \lambda\|\mathbf{S}\|_1 \\ + \langle \mathbf{Y}, \mathbf{D}-\mathbf{L}-\mathbf{S}\rangle \\ + \frac{\mu}{2}\|\mathbf{D}-\mathbf{L}-\mathbf{S}\|_F^2,\end{aligned} \tag{2}$$

where $\mathbf{Y} \in \mathbb{R}^{m\times n}$ is the Lagrangian variable and $\mu > 0$ is a penalty parameter. (2) can be solved via alternating directions method, aka solve the problem for each variable separately at each iteration [16]. In this case each resulting subproblem has a closed form solution so that the algorithm iterates as follows:

1. Solve (2) for $\mathbf{S}$ with fixed $\mathbf{L}$ and $\mathbf{Y}$. This is done by element-wise soft thresholding the appropriate intermediate matrix, i.e. $\mathbf{S}^{(k+1)} = \mathcal{S}_{\lambda\mu^{-1}}[\mathbf{D} - \mathbf{L}^{(k)} + \mu^{-1}\mathbf{Y}^{(k)}]$, where $\mathcal{S}_\tau[\mathbf{M}]$ is the soft thresholding operator defined element-wise [8]:

$$\mathcal{S}_\tau[x] = (|x| - \tau)_+ \text{sgn}(x). \tag{3}$$

2. Solve (2) for $\mathbf{L}$ with fixed $\mathbf{S}$ and $\mathbf{Y}$ or equivalently solve

$$\min_{\mathbf{L}}\{\|\mathbf{L}\|_* + \frac{\mu}{2}\|\mathbf{M}-\mathbf{L}\|_F^2\}, \tag{4}$$

where $\mathbf{M} = \mathbf{D} - \mathbf{S}_k + \mu^{-1}\mathbf{Y}^{(k)}$. This in turn, can be done in closed form using the singular value thresholding operator $\mathcal{D}_\tau[\mathbf{M}]$ [5], i.e. $\mathbf{L}^{(k+1)} = \mathcal{D}_{\mu^{-1}}[\mathbf{D} - \mathbf{S}^{(k+1)} + \mu^{-1}\mathbf{Y}^{(k)}]$, where

$$\mathcal{D}_\tau[\mathbf{M}] = \mathbf{U}\mathcal{S}_\tau[\Sigma]\mathbf{V}^\top, \tag{5}$$

where $\mathbf{M} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$ is the SVD of $\mathbf{M}$.

3. Update Lagrange variables $\mathbf{Y}$ and penalty coefficients μ.

This procedure was dubbed Inexact ALM (IALM) in [16] and is formally given here in Algorithm 1. Note that for practical efficiency only a few singular values are computed as suggested in [16].

2.2. Non-convex RPCA

In a recent paper Netrapalli et al proposed a new method for recovering a low-rank matrix from sparse corruptions [22]. Its main idea is to perform alternating projects onto low rank and sparse matrix spaces. Although these sets are not convex, projections onto them can be done efficiently

Algorithm 1 Inexact ALM (IALM)

Input: $\mathbf{D} \in \mathbb{R}^{m \times n}$

for $k \leftarrow 1$ to ... **do**
  // Solve $\arg\min_{\mathbf{S}} \mathcal{L}(\mathbf{L}^{(k+1)}, \mathbf{S}, \mathbf{Y}^{(k)}, \mu_k)$
  $\mathbf{S}^{(k+1)} \leftarrow \mathcal{S}_{\lambda\mu_k^{-1}}[\mathbf{D} - \mathbf{L}^{(k+1)} + \mu_k^{-1}\mathbf{Y}^{(k)}]$
  // Solve $\arg\min_{\mathbf{L}} \mathcal{L}(\mathbf{L}, \mathbf{S}^{(k+1)}, \mathbf{Y}^{(k)}, \mu_k)$
  $\mathbf{M} \leftarrow \mathbf{D} - \mathbf{S}^{(k+1)} + \mu_k^{-1}\mathbf{Y}^{(k)}$
  $(\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}) \leftarrow \text{SVD}(\mathbf{M})$
  $\mathbf{L}^{(k+1)} \leftarrow \mathbf{U}\mathcal{S}_{\mu_k^{-1}}[\mathbf{\Sigma}]\mathbf{V}^\top$
  // Update the Lagrangian variable
  $\mathbf{Y}^{(k+1)} \leftarrow \mathbf{Y}^{(k)} + \mu_k(\mathbf{D} - \mathbf{L}^{(k+1)} - \mathbf{S}^{(k+1)})$
  Update $\mu_k \leftarrow \mu_{k+1}$
end forreturn $(\mathbf{L}^{(k+1)}, \mathbf{S}^{(k+1)})$

using the hard thresholding operator $\mathcal{H}_\zeta[x]$, which is applied on vectors and matrices element-wise, i.e. $\mathcal{H}_\zeta[\mathbf{X}]_{i,j} = \mathbf{X}_{i,j}$ if $|\mathbf{X}_{i,j}| \geq \zeta$ and 0 otherwise. Specifically, solve the following non-convex problem of finding a low rank plus sparse matrix decomposition:

$$\min_{\mathbf{L}} \|\mathbf{D} - \mathbf{L}\|_0, \quad \textit{subject to} \quad \text{rank}(\mathbf{L}) \leq r, \tag{6}$$

where r is a given upper bound on the rank of low rank component $\mathbf{L}^\star$. Although the problem in (6) is not convex, it can provably be solved in linear time under mild conditions. The main steps of the method are given in Algorithm 2, which alternatively solves two sub-problems at each iteration by fixing one variable and solving for the other. The main difference here is that the arising sub-problems are constrained with the corresponding non-convex sets, therefore hard thresholding is used instead of soft thesholding.

Here $\text{SVD}(\mathbf{M}, l)$ returns the first l singular values with corresponding singular vectors. Although Algorithm 2 requires only l singular values at stage $l = 1, \ldots, r$, SVD operations are still the computational bottleneck.

3. Multilevel Approximate SVD

In this section, we present a simple and yet efficient method for calculating SVD of a matrix $\mathbf{M} \in \mathbb{R}^{m \times n}$ inspired from multilevel optimization algorithms [20, 13]. The main idea is to first create a lower dimensional coarse matrix and then calculate its SVD, which is then used for low rank approximations.

As in multilevel optimisation algorithms our method too uses so-called "restriction" operators to construct coarse models. We denote the restriction operator as $\mathbf{R}$ and assume that it has linearly independent columns.

Algorithm 2 Alternating Projections (AltProj)

Input: $\mathbf{D} \in \mathbb{R}^{m \times n}$, target rank r

Initialize $\mathbf{L}^{(0)} = \mathbf{0}$ and $\mathbf{S}^{(0)} = \mathcal{H}_{\zeta_0}(\mathbf{D} - \mathbf{L}^{(0)})$
for Stage $l \leftarrow 1$ to r **do**
  for Iteration $k \leftarrow 0$ to T **do**
    // Solve $\arg\min_{\mathbf{L}:\text{rank}(\mathbf{L}) \leq l} \|\mathbf{D} - \mathbf{L} - \mathbf{S}^{(k)}\|_2^2$
    $\mathbf{M} \leftarrow \mathbf{D} - \mathbf{S}^{(k)}$
    $(\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}) \leftarrow \text{SVD}(\mathbf{M}, l)$
    $\mathbf{L}^{(k+1)} \leftarrow \mathbf{U}\mathcal{H}_l[\mathbf{\Sigma}]\mathbf{V}^\top$
    // Solve $\arg\min_{\mathbf{S}:\|\mathbf{S}\|_0 \leq \zeta} \|\mathbf{D} - \mathbf{L}^{(k+1)} - \mathbf{S}\|_2^2$
    Update threshold ζ as in [22]
    $\mathbf{S}^{(k+1)} \leftarrow \mathcal{H}_\zeta[\mathbf{D} - \mathbf{L}^{(k+1)}]$
    if $\sigma_{l+1}(\mathbf{L}^{(k+1)}) < \epsilon$ **then**
      return $(\mathbf{L}^{(T)}, \mathbf{S}^{(T)})$
    else
      $\mathbf{S}^{(0)} \leftarrow \mathbf{S}^{(T)}$
    end if
  end for
end for
return $(\mathbf{L}^{(T)}, \mathbf{S}^{(T)})$

Specifically, for $\mathbf{R} \in \mathbb{R}^{n \times \frac{n}{2}}$ we use

$$\mathbf{R}_n = \frac{1}{4}\begin{bmatrix} \alpha & 0 & 0 & \ldots & 0 & 0 \\ 4-2\alpha & 0 & 0 & \ldots & 0 & 0 \\ \alpha & \alpha & 0 & \ldots & 0 & 0 \\ 0 & 4-2\alpha & 0 & \ldots & 0 & 0 \\ 0 & \alpha & \alpha & \ldots & 0 & 0 \\ & & & \ldots & & \\ 0 & 0 & 0 & \ldots & \alpha & \alpha \\ 0 & 0 & 0 & \ldots & 0 & 4-2\alpha \\ 0 & 0 & 0 & \ldots & 0 & \alpha \end{bmatrix} \tag{7}$$

for some $0 \leq \alpha \leq 1$ acting as a smoothing parameter. For instance, when $\alpha = 1$, $\mathbf{R}_n$ is the standard interpolation operator [4], and when $\alpha = 0$, $\mathbf{R}_n$ simply selects every other column of $\mathbf{M}$.

Often in practice we use more than 2 levels of coarse models. Specifically, we use a restriction operator $\mathbf{R} = \mathbf{R}_n\mathbf{R}_{\frac{n}{2}} \ldots \mathbf{R}_{n_H} \in \mathbb{R}^{m \times n_H}$, where $\mathbf{R}_k \in \mathbb{R}^{k \times \frac{k}{2}}$ is given as in (7). For all experiments we used up to the deepest possible levels, so that $n_H > r$, where r is the number of singular values required by the overlying algorithm. Clearly, $\mathbf{R}$ has linearly independent columns and thus is full rank, moreover, without loss of generality we can assume that $\mathbf{R}$ has normalized columns so that $\|\mathbf{R}\|_2 \leq 1$.

Next, we use the restriction operator $\mathbf{R}$ to present our proposed CoarseSVD method for efficiently calculating approximate SVDs in Algorithm 3. The basic idea here is to first apply the restriction operator on $\mathbf{M}$ and then perform SVD on the lower dimensional coarse model.

Algorithm 3 CoarseSVD

Input: $\mathbf{M} \in \mathbb{R}^{m\times n}$, $\mathbf{R} \in \mathbb{R}^{n\times n_H}$

$\mathbf{M}_H \leftarrow \mathbf{MR}$

$(\mathbf{U}_H, \mathbf{\Sigma}_H, \mathbf{V}_H) \leftarrow \text{SVD}(\mathbf{M}_H)$

return $(\mathbf{U}_H, \mathbf{\Sigma}_H, \mathbf{V}_H)$

Then after finding a low rank approximation for $\mathbf{M}$, we "lift" it to the original fine problem dimension. As in multilevel literature, this is done using the transpose of the restriction operator. Proposition 1 characterizes the approximate SVD after prolongation.

Proposition 1. *The prolongation* $\mathbf{V}^H = \mathbf{V}_H\mathbf{R}^\top$ *of right singular vectors satisfies*

1. $\mathbf{V}^{H\top}\mathbf{V}^H = \mathbf{R}\mathbf{R}^T$
2. $\|\mathbf{V}^H\mathbf{V}^{H\top}\| = \|\mathbf{R}^\top\mathbf{R}\|$
3. $\|\mathbf{U}_H\mathbf{\Sigma}_H\mathbf{V}_H^\top\mathbf{R}^\top - \mathbf{M}\| \le \delta\|\mathbf{M}\|$,

for a constant $\delta > 0$ *and either* $\|\cdot\|_2$ *and* $\|\cdot\|_F$ *norms.*

Proof. The first two results follow directly from the orthogonality of $\mathbf{V}_H$. The third one is due to the fact that $\|\mathbf{M}(\mathbf{R}\mathbf{R}^\top - \mathbf{I})\| \le \delta\|\mathbf{M}\|$, for

$$\delta := \|\mathbf{R}\mathbf{R}^\top - \mathbf{I}\| \le 1. \tag{8}$$

□

4. Multilevel Algorithms via Coarse SVD

While both Inexact ALM and Alternating Projections algorithms are guaranteed to converge to the global optima with strong convergence rates, in practice performing SVDs still remains a computational bottleneck, especially for larger problems. To overcome this major obstacle we propose to instead construct lower dimensional counterparts for the computationally expensive parts of each algorithm and use their solutions for finding approximate solutions for the original fine level problems. In other words, we use the multilevel low rank approximation technique to accelerate both convex and non-convex robust PCA algorithms.

4.1. Multilevel Inexact ALM

We begin this subsection by introducing the multilevel singular value thresholding (ML-SVT) operator defined as

$$\mathcal{D}_\tau^H[\mathbf{M}] = \mathbf{U}_H\mathcal{S}_\tau[\mathbf{\Sigma}_H]\mathbf{V}_H^\top\mathbf{R}^\top, \tag{9}$$

where $\mathbf{MR} = \mathbf{U}_H\mathbf{\Sigma}_H\mathbf{V}_H^\top$ is the SVD of the coarse model $\mathbf{M}_H = \mathbf{MR}$. Note that the ML-SVT operator (9) requires a SVD on a $m \times n_H$ matrix - significantly cheaper than the $m \times n$ for (5). The next theorem shows that the proposed ML-SVT operator gives a good approximate solution for problem (4).

Theorem 1. *Assuming that* $\|\mathbf{R}\|_2 \le 1$ *and* $0 < \tau \le \sigma_{H,1}$, *the ML-SVT operator* $\mathcal{D}_\tau^H[\mathbf{M}]$ *gives a* $\frac{\sigma_{H,1}}{\tau^2}(\sigma_1 + \sigma_{H,1} - \tau)$*-approximate solution to the problem (4), where* $\sigma_{H,1}$ *is the largest singular value of* $\mathbf{MR}$.

Proof. The proof follows the steps of the proof of Theorem 2.1 of [5] and is given in the Appendix. □

Then we can use the proposed ML-SVT operator to solve the corresponding subproblems of Algorithm (1) resulting the Multilevel Inexact ALM (ML-IALM) method given below as Algorithm 4.

Algorithm 4 Multilevel Inexact ALM (ML-IALM)

Input: $\mathbf{D}, \mathbf{S}^{(0)}, \mathbf{Y}^{H,(0)} \in \mathbb{R}^{m\times n}$; μ_0

for $k \leftarrow 1$ to ... **do**

  $\mathbf{S}^{(k+1)} \leftarrow \mathcal{S}_{\lambda\mu_k^{-1}}[\mathbf{D} - \mathbf{L}^{H(k+1)} + \mu_k^{-1}\mathbf{Y}^{(k)}]$

  // Approx solve $\arg\min_{\mathbf{L}} \mathcal{L}(\mathbf{L}, \mathbf{S}^{(k+1)}, \mathbf{Y}^{(k)}, \mu_k)$

  $\mathbf{M}_H \leftarrow (\mathbf{D} - \mathbf{S}^{(k+1)} + \mu_k^{-1}\mathbf{Y}^{(k)})\mathbf{R}$

  $(\mathbf{U}_H, \mathbf{\Sigma}_H, \mathbf{V}_H) \leftarrow \text{SVD}(\mathbf{M}_H)$

  $\mathbf{L}_H \leftarrow \mathbf{U}_H\mathcal{S}_{\mu^{-1}}[\mathbf{\Sigma}_H]\mathbf{V}_H^\top$

  $\mathbf{L}^{H(k+1)} \leftarrow \mathbf{L}_H\mathbf{R}^\top$

  // Continue as in Algorithm 1

  $\mathbf{Y}^{(k+1)} \leftarrow \mathbf{Y}^{(k)} + \mu_k(\mathbf{D} - \mathbf{L}^{H(k+1)} - \mathbf{S}^{(k+1)})$

  Update $\mu_k \leftarrow \mu_{k+1}$

end for

We finish this subsection with a remark that the same approach can be used to extend the Inexact ALM method for the more general matrix completion problem (Algorithm 6 in [16]). In this case the only difference is that instead of soft-thresholding, projection onto a simple convex space Ω is used to update $\mathbf{S}^{(k+1)}$, whereas updates for $\mathbf{L}^{(k+1)}$ are the same and therefore multilevel SVD can be used.

4.2. Multilevel Alternating Projections

We apply the multilevel low rank approximation method of Algorithm 3 also within the non-convex alternating projections algorithm. In this case we create coarse models for subproblems of finding low rank approximations for intermediate matrices $\mathbf{M} = \mathbf{D} - \mathbf{S}^{(k)}$, solve these subproblems and lift their solutions to the original fine dimension.

Each iteration of the Alternating Projections algorithm requires solving

$$\min_{\mathbf{L}\in\mathbb{R}^{m\times n}} \|\mathbf{D} - \mathbf{L} - \mathbf{S}\|_2 \quad s.t. \quad \text{rank}(\mathbf{L}) \le l, \tag{10}$$

which has a closed form solution given by the hard thresholding operator as follows

$$\hat{\mathbf{L}} = \mathbf{U}\mathcal{H}_l[\mathbf{\Sigma}]\mathbf{V}^\top, \tag{11}$$

where $\mathcal{H}_l[x]$ is the hard thresholding operator and is defined element-wise for vectors and matrices. Therefore, for this setting we use the hard thresholding operator on the coarse model to construct an approximate solution for problem (10) as follows:

$$\mathbf{L}^H = \mathbf{U}_H \mathcal{H}_l[\mathbf{\Sigma}_H] \mathbf{V}_H^\top \mathbf{R}^\top, \tag{12}$$

where $\mathbf{MR} = \mathbf{M}_H = \mathbf{U}_H \mathbf{\Sigma}_H \mathbf{V}^\top$ is a SVD of the coarse model. Notice that the multilevel operator computes SVDs on much smaller problems than the original algorithm in this case as well. Then in Theorem 2 we show that the $\mathbf{L}^H$ defined in (12) gives a good approximate solution for problem (10).

Theorem 2. *The multilevel low rank approximation procedure of (12) gives a $(\sigma_1 + \sigma_{H,1})$-approximate solution of the problem (10), where $\sigma_{H,1}$ is the largest singular value of $\mathbf{M}_H = \mathbf{MR}$.*

Proof. The proof follows the proof of Eckhard-Young-Mirsky theorem [11] and is given in the Appendix. □

Then we can use (12) inside Algorithm 2 to efficiently solve the corresponding subproblem. The resulting method is presented in Algorithm 5.

Algorithm 5 Multilevel Alternating Projections (ML-AltProj)

Input: $\mathbf{D} \in \mathbb{R}^{m \times n}$, target rank r

Initialize $\mathbf{L}^{H^{(0)}} = \mathbf{0}$ and $\mathbf{S}^{(0)} = \mathcal{H}_{\zeta_0}(\mathbf{D} - \mathbf{L}^{H^{(0)}})$
for Stage $l \leftarrow 1$ to r **do**
  for Iteration $k \leftarrow 0$ to T **do**
    // Approx solve $\underset{\mathbf{L}:\text{rank}(\mathbf{L}) \le l}{\arg\min} \|\mathbf{D} - \mathbf{L} - \mathbf{S}^{(k)}\|_2$
    $\mathbf{M}_H \leftarrow (\mathbf{D} - \mathbf{S}^{(k)})\mathbf{R}$
    $(\mathbf{U}_H, \mathbf{\Sigma}_H, \mathbf{V}_H) \leftarrow \text{SVD}(\mathbf{M}_H, l)$
    $\mathbf{L}_H \leftarrow \mathbf{U}_H \mathcal{H}_l[\mathbf{\Sigma}_H] \mathbf{V}_H^\top$
    $\mathbf{L}^{H^{(k+1)}} \leftarrow \mathbf{L}_H \mathbf{R}^\top$
    // Continue as in Algorithm 2
    Update threshold ζ as in [22]
    $\mathbf{S}^{(k+1)} \leftarrow \mathcal{H}_\zeta[\mathbf{D} - \mathbf{L}^{H^{(k+1)}}]$
    if $\sigma_{l+1}(\mathbf{L}^{H^{(k+1)}}) < \epsilon$ **then**
      return $(\mathbf{L}^{H^{(T)}}, \mathbf{S}^{(T)})$
    else
      $\mathbf{S}^{(0)} \leftarrow \mathbf{S}^{(T)}$
    end if
  end for
end for
return $(\mathbf{L}^{H^{(T)}}, \mathbf{S}^{(T)})$

5. Experiments

To test the practical efficiency of the proposed methods we compare them with the standard Inexact ALM [16]and Alternating Projections [22] algorithms on several real life video background extraction and facial shadow removal problems. For the standard Inexact ALM and Alternating Projections algorithms we used the provided Matlab code. Then for each multilevel variant we replaced the standard low rank approximation parts of respective algorithms with corresponding multilevel low rank approximation code, keeping the rest of the algorithms unchanged. Particularly, we used the same optimality criteria, so that the comparisons are fair. All methods were tested in Matlab R2015a on a standard desktop machine with Intel Core i7 processor and 32GB RAM.

5.1. Video Background Extraction

First, we test the algorithms on real surveillance videos. Assume we are given a surveillance video from a fixed camera and the task is to separate the constant background from moving objects. This problem can be modeled as (convex or non-convex) RPCA [3]. We first stack each frame of the video as a column vector creating a data matrix $\mathbf{D}$. Then, since the fixed background remains (approximately) constant in each frame and the moving objects take a relatively small portion of each frame, they can respectively represent the low rank and sparse components of the RPCA decomposition. We tested all algorithms on 4 surveillance videos described below:

- **highway**: $48 \times 64 \times 400$; run 2 seconds
- **copy machine**: $48 \times 72 \times 3400$; run 50 seconds
- **walk:** $240 \times 320 \times 794$; run 50 seconds [28]
- **gates:** $240 \times 320 \times 1895$; run 200 seconds [28]

The results are reported in Figure 1. Each row represents a tested video. The first column contains sample frames from each corresponding video, then each of the following column triplets contains corresponding low rank and sparse components as returned from IALM and ML-IALM and ML-AltProj algorithms. We run IALM and ML-IALM for the same fixed time and ML-IALM until convergence with 10^{-7} error for reference. Below each frame we also report the corresponding achieved rank and the feasibility gap (FG) i.e. $\|\mathbf{D} - \mathbf{L}^\star - \mathbf{S}^\star\|_F / \|\mathbf{D}\|_F$. As the results indicate, all algorithms produce similar results for all videos, except **copymachine**, for which ML-IALM produces significantly clearer separation of background than IALM. This is because **copy machine** has largest number of frames relative to the frame dimension.

We do not present frame samples from AltProj since it returns visually similar values to ML-AltProj. In this case,

Original	Low Rank			Sparse		
	IALM	ML-IALM	ML-AltProj	IALM	ML-IALM	ML-AltProj
						
highway	rank =5	3	1	FG = 0.0176	0.01	$3.6 \cdot 10^{-4}$
						
copy machine	rank = 7	4	1	FG = 0.0364	0.0031	$2.8 \cdot 10^{-4}$
						
walk	rank = 2	1	1	FG = 0.02	0.0231	$4 \cdot 10^{-4}$
						
gates	rank = 3	3	2	FG = 0.05	0.04	$2.8 \cdot 10^{-4}$

Figure 1: Examples from solving video background extraction problems via IALM, ML-IALM and ML-AltProj methods. IALM and ML-IALM run for a fixed CPU seconds, while ML-AltProj is given for reference and runs until convergence error 10^{-7}. Each row corresponds respectively to **highway** ($48 \times 64 \times 400$), **copy machine** ($48 \times 72 \times 3,400$), **walk** ($240 \times 320 \times 794$) and **gates** ($240 \times 320 \times 1,895$) videos from top to bottom. With each frame we also report the respective rank of the low rank component and the feasibility gap (FG): $\|\mathbf{D} - \mathbf{L} - \mathbf{S}\|_F / \|\mathbf{D}\|_F$.

Problem	AltProj	ML-AltProj
highway	7	3 (2 levels)
copy machine	54	12 (8 levels)
walk	467	261 (6 levels)
gates	744	354 (8 levels)

Table 1: CPU times (in seconds) after solving video background removal problems up to tolerance 10^{-7} using the standard non-convex alternating projections algorithm and its multilevel variant.

we ran each algorithm until the same optimality error and report CPU times (in seconds) in Table 1. In all experiments we used up to 4-7 levels of coarse models (more for larger problems). In this case as well, the multilevel variant largely outperforms the original algorithm. In fact, the larger the original problem, the bigger relative speed up can be achieved using the multilevel approach, since for larger n we can use deeper levels.

For further investigation of the convergence properties of IALM and AltProj algorithms compared to their multilevel variants, we measure the relative error of the current iterates compared to the ground truth $(\mathbf{L}_0, \mathbf{S}_0)$ and FGs during the iterations of both standard and multilevel algorithms through the same time interval. We report those relative errors against CPU time (seconds) and iteration numbers in Figure 2.

The plots suggest that ML-IALM performs only slightly faster than IALM on the smaller **highway** example, however, as expected it is significantly faster on the larger **copy machine** and **gates** problems. As we could anticipate from the theory, at each iteration ML-IALM achieves a very good approximation as measured by the reconstruction error, and since its iterations are significantly cheaper, it performs more iterations during the same time interval than IALM. We also report the results of running AltProj and ML-AltProj methods on the **walk** problem. The results are very similar to those observed in the convex model. ML-AltProj decreases the relative errors much earlier during the iterations and has significantly cheaper per iteration complexity.

5.2. Shadow Removal from Facial Images

Here we have a set of facial images from one or more individuals under various illuminations and the task is to remove shadow/light noises from images. We used images of individuals from the Yale B facial extended database [10]. It contains (96×84) facial images of 39 subjects taken under various poses and illuminations each, with total $2,414$ images.

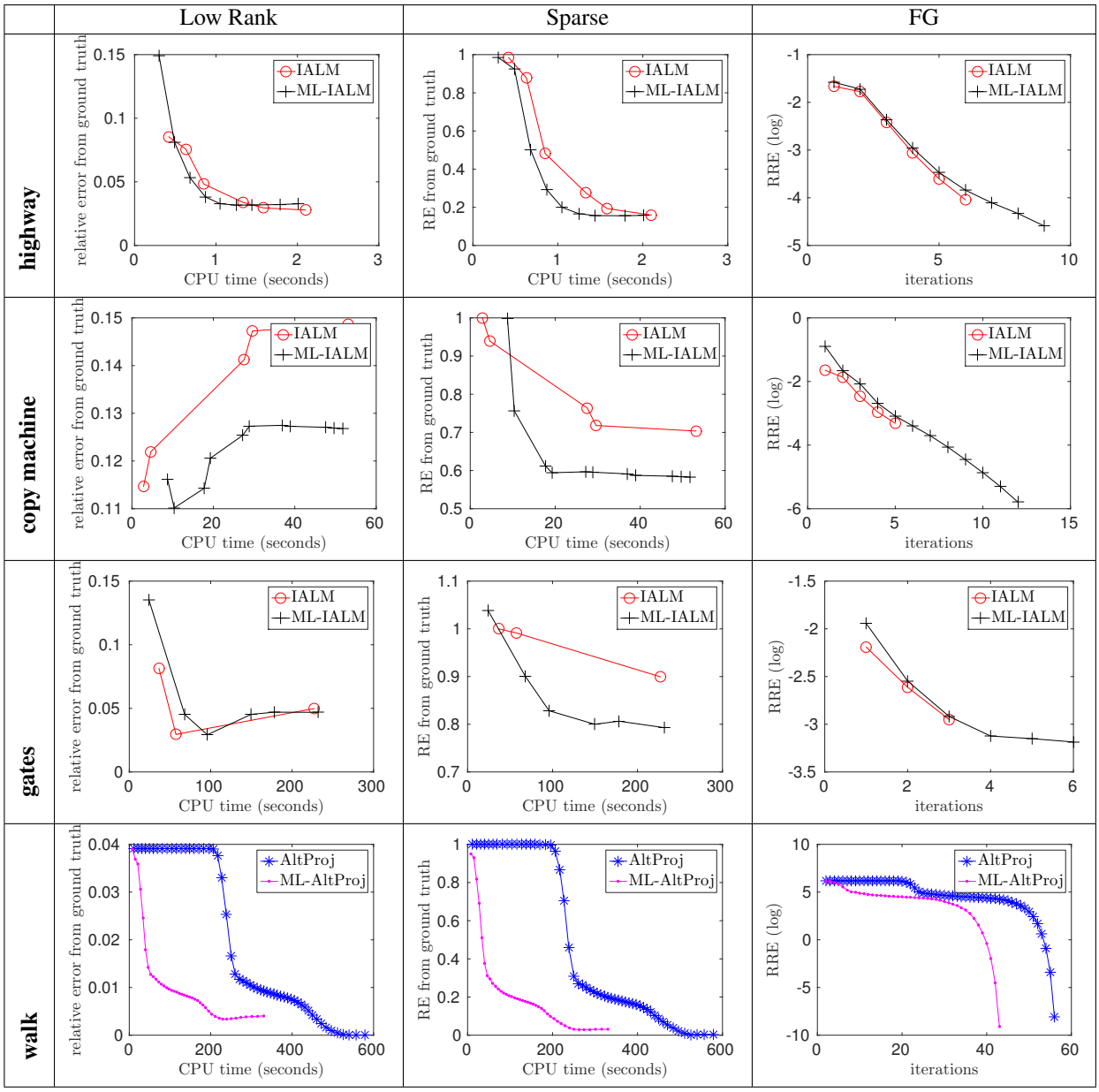


Figure 2: Comparing the relative errors during IALM, ML-IALM, AltProj and ML-AltProj iterations. The first two columns give relative errors compared to the ground truth $(\mathbf{L}_0, \mathbf{S}_0)$, and the third column gives FGs during iterations. Each row corresponds respectively to **highway** $(48 \times 64 \times 400)$, **copy machine** $(48 \times 72 \times 3,400)$, **walk** $(240 \times 320 \times 794)$ and **gates** $(240 \times 320 \times 1,895)$ videos from top to bottom.

For this setting as well we test the multilevel SVD plugged into both IALM and AltProj methods. We ran both IALM and ML-IALM algorithms for 5 second and compare the returned results. While AltProj and ML-AltProj run until convergence with 10^{-7} error. The results are reported in Figure 3. Here as well each row represents a particular database (individual). The first column contains sample frames from each corresponding facial database, then each of the following four columns contains correspondingly low rank and sparse components as returned from Inexact IALM, ML-IALM, AltProj and ML-AltProj algorithms. With each image we also report the corresponding achieved rank of the low rank component and the FG.

In order to compare the multilevel approach for the non-convex problem, we report the results after running AltProj and ML-AltProj algorithms until achieving convergence er-

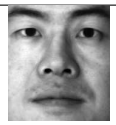
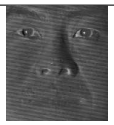
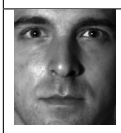
Original	Low Rank				Sparse			
	IALM	ML-IALM	AltProj	ML-AltProj	IALM	ML-IALM	AltProj	ML-AltProj
								
Yale B01	rank = 5	10	10	10	FG = 0.24	0.05	$4 \cdot 10^{-4}$	10^{-4}
								
Yale B02	rank = 5	10	10	10	FG = 0.24	0.05	$4 \cdot 10^{-4}$	10^{-4}
								
Yale B10	rank = 3	10	10	10	FG = 0.24	0.05	$4 \cdot 10^{-4}$	10^{-4}

Figure 3: Examples from solving facial shadow removal problems via IALM, ML-IALM, AltProj and ML-AltProj algorithms on cropped **Yale B** database ($96 \times 84 \times 2414$). We run both IALM and ML-IALM for fixed five seconds, while AltProj and ML-AltProj run until convergence with 10^{-7} error. With each image we also report the respective rank of the low rank component and the feasibility gap (FG): $\|\mathbf{D} - \mathbf{L} - \mathbf{S}\|_F / \|\mathbf{D}\|_F$.

Problem	AltProj	Ml-AltProj
Yale B01	38.8	15.3
Yale B02	39.1	15.2
Yale B10	37.4	16.8

Table 2: CPU times (in seconds) after solving shadow removal problems up to a fixed tolerance using the standard non-convex alternating projections algorithm and its multilevel variant. For all experiments we used 2 levels for the multilevel algorithm.

ror 10^{-7} and record CPU times (seconds). The results are reported in Table 2. In all experiments we used up to 3 levels of coarse models. In all experiments the multilevel algorithm is more than twice faster than its standard counterpart.

6. Discussion

In this paper we presented an approximate multilevel dimension reduction method for efficient SVD calculations. We showed that the multilevel algorithms are theoretically good approximations to the original problem, and moreover, in practice they are **significantly** faster as demonstrated within convex and non-convex RPCA models. Finally, its applications can be extended further to other SVD based methods, such as RASL [25] and matrix completion [7]. More interesting application of multilevel methods would be their application to tensor decomposition problems (see [15] and references therein).

References

[1] Z. Allen-Zhu and Y. Li. Even faster svd decomposition yet without agonizing pain. In *Advances in Neural Information Processing Systems*, pages 974–982, 2016. 2

[2] R. Angst, C. Zach, and M. Pollefeys. The generalized trace-norm and its application to structure-from-motion problems. In *2011 International Conference on Computer Vision*, pages 2502–2509. IEEE, 2011. 1

[3] T. Bouwmans, A. Sobral, S. Javed, S. K. Jung, and E.-H. Zahzah. Decomposition into low-rank plus additive matrices for background/foreground separation: A review for a comparative evaluation with a large-scale dataset. *Computer Science Review*, 2016. 5

[4] W. L. Briggs, S. F. McCormick, et al. *A multigrid tutorial*. Siam, 2000. 3

[5] J.-F. Cai, E. J. Candès, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956–1982, 2010. 2, 4

[6] E. J. Candès, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):11, 2011. 1

[7] E. J. Candès and B. Recht. Exact matrix completion via convex optimization. *Foundations of Computational mathematics*, 9(6):717–772, 2009. 8

[8] I. Daubechies, M. Defrise, and C. De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on pure and applied mathematics*, 57(11):1413–1457, 2004. 1, 2

[9] P. Drineas, R. Kannan, and M. W. Mahoney. Fast monte carlo algorithms for matrices ii: Computing a low-rank approximation to a matrix. *SIAM Journal on computing*, 36(1):158–183, 2006. 2

[10] A. Georghiades, P. Belhumeur, and D. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Trans. Pattern Anal. Mach. Intelligence*, 23(6):643–660, 2001. 6

[11] G. H. Golub and C. F. Van Loan. *Matrix computations*, volume 3. JHU Press, 2012. 5

[12] H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, 24(6):417, 1933. 1

[13] V. Hovhannisyan, P. Parpas, and S. Zafeiriou. Magma: Multilevel accelerated gradient mirror descent algorithm for large-scale convex composite minimization. *SIAM Journal on Imaging Sciences*, 9(4):1829–1857, 2016. 2, 3

[14] P. J. Huber. *Robust statistics*. Springer, 2011. 1

[15] J. Kossaifi, Y. Panagakis, and M. Pantic. Tensorly: Tensor learning in python. In *NIPS Tensor-Learn Workshop*, 2016. 8

[16] Z. Lin, M. Chen, and Y. Ma. The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. *arXiv preprint arXiv:1009.5055*, 2010. 1, 2, 4, 5

[17] G. Liu and S. Yan. Active subspace: Toward scalable low-rank learning. *Neural computation*, 24(12):3371–3394, 2012. 1

[18] R. Liu, Z. Lin, Z. Su, and J. Gao. Linear time principal component pursuit and its extensions using l1 filtering. *Neurocomputing*, 142:529–541, 2014. 2

[19] C. Musco and C. Musco. Randomized block krylov methods for stronger and faster approximate singular value decomposition. In *Advances in Neural Information Processing Systems*, pages 1396–1404, 2015. 2

[20] S. Nash. A multigrid approach to discretized optimization problems. *Optimization Methods and Software*, 14(1-2):99–116, 2000. 2, 3

[21] B. K. Natarajan. Sparse approximate solutions to linear systems. *SIAM journal on computing*, 24(2):227–234, 1995. 1

[22] P. Netrapalli, U. Niranjan, S. Sanghavi, A. Anandkumar, and P. Jain. Non-convex robust pca. In *Advances in Neural Information Processing Systems*, pages 1107–1115, 2014. 1, 2, 3, 5

[23] T.-H. Oh, Y. Matsushita, Y.-W. Tai, and I. So Kweon. Fast randomized singular value thresholding for nuclear norm minimization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4484–4493, 2015. 2

[24] N. Parikh and S. Boyd. Proximal algorithms. *Foundations and Trends in optimization*, 1(3):123–231, 2013. 1

[25] Y. Peng, A. Ganesh, J. Wright, W. Xu, and Y. Ma. RASL: Robust alignment by sparse and low-rank decomposition for linearly correlated images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(11):2233–2246, 2012. 1, 8

[26] C. Sagonas, Y. Panagakis, S. Zafeiriou, and M. Pantic. Raps: Robust and efficient automatic construction of person-specific deformable models. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1789–1796. IEEE, 2014. 1

[27] C. Sagonas, Y. Panagakis, S. Zafeiriou, and M. Pantic. Robust statistical frontalization of human and animal faces. *International Journal of Computer Vision*, pages 1–22, 2016. 1

[28] A. Vacavant, T. Chateau, A. Wilhelm, and L. Lequièvre. A benchmark dataset for outdoor foreground/background extraction. In *Asian Conference on Computer Vision*, pages 291–300. Springer, 2012. 5

[29] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM review*, 38(1):49–95, 1996. 1

[30] Z. Zhang, A. Ganesh, X. Liang, and Y. Ma. Tilt: Transform invariant low-rank textures. *International Journal of Computer Vision*, 99(1):1–24, 2012. 1